Data Types and Variables

Objectives:

- Understand the various data types in Java.
- Learn how to declare and use different types of variables.

Java Basic Data Types

Java provides different types of data types to handle different kinds of values. These are broadly classified into two categories:

1. **Primitive Data Types**
2. **Reference Data Types**

1. Primitive Data Types

Primitive data types are the most basic types of data. Java has eight primitive data types:

Data Type	Size	Default Value	Description
`byte`	1 byte	0	Stores small integer values from -128 to 127
`short`	2 bytes	0	Stores integer values from -32,768 to 32,767
`int`	4 bytes	0	Stores integer values from -2^31 to (2^31)-1
`long`	8 bytes	0L	Stores large integer values from -2^63 to (2^63)-1
`float`	4 bytes	0.0f	Stores fractional numbers, sufficient for 6-7 decimal digits
`double`	8 bytes	0.0d	Stores fractional numbers, sufficient for 15 decimal digits
`char`	2 bytes	'\u0000'	Stores a single character
`boolean`	1 bit	false	Stores true or false values

Example: Implementing All Primitive Data Types

```java
public class PrimitiveDataTypes {
    public static void main(String[] args) {
        byte a = 10;
        short b = 2000;
        int c = 50000;
        long d = 15000000000L;
        float e = 5.75f;
        double f = 19.99;
        char g = 'A';
        boolean h = true;
        System.out.println("byte: " + a);
        System.out.println("short: " + b);
        System.out.println("int: " + c);
        System.out.println("long: " + d);
        System.out.println("float: " + e);
        System.out.println("double: " + f);
        System.out.println("char: " + g);
        System.out.println("boolean: " + h);
    }
}
```

2. Reference Data Types

Reference data types store references to memory locations rather than actual values. The most common reference type is `String`, but objects and arrays also fall under this category.

String Data Type

A `String` is a sequence of characters enclosed in double quotes.

Example:

```
public class StringExample {
    public static void main(String[] args) {
        String message = "Hello, Java!";
        System.out.println(message);
    }
}
```

Array Data Type

An array is a collection of elements of the same type stored in contiguous memory locations.

Example:

```
public class ArrayExample {
    public static void main(String[] args) {
        int[] numbers = {1, 2, 3, 4, 5};
        System.out.println("First element: " + numbers[0]);
    }
}
```

Java Literals

Literals are constant values assigned to variables. Java supports the following types of literals:

- **Integer Literals**: `int num = 10;`
- **Floating-Point Literals**: `double pi = 3.1415;`
- **Character Literals**: `char letter = 'A';`
- **String Literals**: `String greeting = "Hello";`
- **Boolean Literals**: `boolean flag = true;`

Example:

```
public class LiteralExample {
    public static void main(String[] args) {
        int decimal = 100;
        int binary = 0b1101;
        int octal = 0123;
        int hex = 0x1A;
        float pi = 3.14f;
        char letter = 'J';
        boolean isJavaFun = true;

        System.out.println("Decimal: " + decimal);
        System.out.println("Binary: " + binary);
        System.out.println("Octal: " + octal);
        System.out.println("Hexadecimal: " + hex);
        System.out.println("Float: " + pi);
        System.out.println("Character: " + letter);
        System.out.println("Boolean: " + isJavaFun);
    }
}
```

Java Variable Types

Variables in Java can be classified into three main types:

1. Local Variables

- Declared inside a method, constructor, or block.
- Scope is limited to the method in which it is declared.
- Does not have a default value; must be initialized.

Example:

```
public class LocalVariableExample {
    public void show() {
        int localVar = 5; // Local variable
        System.out.println("Local Variable: " + localVar);
    }
    public static void main(String[] args) {
        LocalVariableExample obj = new LocalVariableExample();
        obj.show();
    }
}
```

2. Instance Variables

- Declared inside a class but outside any method.
- Associated with an object of the class.
- Have default values.

Example:

```java
public class InstanceVariableExample {
    int instanceVar = 10; // Instance variable
    public static void main(String[] args) {
        InstanceVariableExample obj = new InstanceVariableExample();
        System.out.println("Instance Variable: " + obj.instanceVar);
    }
}
```

3. Class/Static Variables

- Declared with the `static` keyword inside a class.
- Shared among all instances of the class.
- Initialized only once at class loading time.

Example:

```java
public class StaticVariableExample {
    static int staticVar = 50; // Static variable
    public static void main(String[] args) {
        System.out.println("Static Variable: " + staticVar);
    }
}
```

Simple Activity

Task: Create a Java program that declares and initializes all primitive and reference data types, then prints their values.

Instructions:

1. Declare and initialize variables for all 8 primitive data types.
2. Create a `String` variable and initialize it.
3. Declare and initialize an array with at least 5 elements.
4. Print all the values to the console.

What is Next?

In the next session, we will discuss **Modifiers and Variables**, covering topics like access modifiers (`public`, `private`, `protected`), final variables, static variables, and transient variables.